SDO 를 통한 디바이스 구성

특수 통신 오브젝트, 소위 **"service data objects"** (SDO) 는 CANopen 장치에 대한 직접적인 액세스에 사용됩니다. 이러한 "service data objects"로, 객체 사전 엔트리들은 읽거나 기록될 수 있으며, 반면 두 개의 노드 (예. 구성되는 노드와 구성될 노드)들 사이는 논리적 1:1 연결 (peer-to-peer)이므로 통신이 항상 발생합니다. 데이터 전송이 승인된 서비스로 이와 같은 연결을 통해 실행된다는 것은, 연결마다 두 개의 CAN 메시지들이 필요하다는 의미이기도 합니다: 네크워크 노드에 요청할 메시지 (SDO request 또는 "Client SDO") 와 노드의 응답용 메시지 (SDO response 또는 "Server SDO"). 여기에 포함된 두 개의 네트워크 노드들은 **SDO client** 나 또는 **SDO server** 라 일컬어집니다; 여기서 서버는 객체 사전(object dictionary)을 통해 데이터를 제공하거나 수용하는 것이고 클라이언트는 데이터를 요청(read)하거나 전송(write)하는 것입니다. 이 두 개의 파트너들 사이에 논리적 연결이 있는데, 이를 **"SDO channel"** 이라 칭합니다. SDO 전송을 위한 개시는 항상 클라이언트에서 비롯됩니다. SDO 전송이 승인되면, 장치가 의미있는 데이터를 제공할 수 없거나 요청 자체가 이미 오류가 있을지라도, 모든 요청이 응답되어야 합니다. 이와 같은 부정적 응답(negative response)을 "abort"라 부릅니다. 이와 같은 응답에서는, abort 의 원인을 명시한, 4-byte long error code (**abort code**) 외에도, 비성공적인 SDO 전송을 언급한 객체 사전 엔트리의 주소 또한 전송됩니다. 이미 설명한대로, SDO 전송은 항상 별도의 프로토콜에 따라 요청-응답 순서로 실행되며, 서비스 데이터 오브젝트의 첫번째 데이터 바이트에 명기됩니다. 따라서 메시지 식별자는 SDO 자체를 지정하며 SDO 의 첫번째 데이터 바이트는 특정 프로토콜을 지정합니다. 이러한 이유로 첫번째 데이터 바이트는 protocol 또는 command byte 라 부르기도 합니다. SDO-message 는 항상 8 byte 길이이며, 필요하지 않은 데이터 필드의 비트는 0 으로 설정되어야 합니다.

어떤 길이의 데이터 필드나 byte 순서라도 객체 사전 액세스를 이용하여 전송될 수 있습니다. 이러한 이유로 SDO 프로토콜을 통해 전송될 수 있는 정보의 길이는 이론적으로는 무제한입니다. SDO 프로토콜은 두 단계로 실행됩니다: 초기 단계에서 객체 사전 엔트리는 주소가 지정되고 전송될 데이터 길이가 공개됩니다. 그러면 두번째 단계는 세그먼트에서 유틸리티 데이터가 전송됩니다 (각각 7 bytes).

이에 따라, DS-301 은 4 개의 서로 다른 SDO 서비스들을 구별합니다: Initiate SDO Upload, Upload SDO Segment, Initiate SDO Download, Download SDO Segment.

매우 흔히 작은 유틸리티 데이터 바이트만 전송될 경우도 있기 때문에, SDO 전송은 단축될 수 있으며 최대 4 bytes 까지 초기화 단계에서 이미 전송될 수 있습니다. 이것을 **"expedited SDO transfer"** 라 부릅니다.

Initiate SDO Download 서비스의 메시지는, CANopen 노드의 객체 사전 엔트리에 대한 write 액세스가 동시에 일어나며, 다음과 같은 구조로 되어 있습니다.

Command byte	OD main-index	OD sub-index	Data (max. 4 bytes)

SDO 서버는 프로토콜 byte 0x60 으로 응답합니다:

Command byte 0x60	OD main-index	OD sub-index	Empty (4 byte)

bit-coded 명령 바이트에서는 서비스가 3 bit (command specifier) 코드로 되어 있습니다. 또 다른 bit 는 expedited 또는 non-expedited 전송이 실행될 것인지를 지정합니다. 추가 bit 는 전송될 데이터 크기가 통신 오브젝트의 마지막 4 byte 에 지정될 것인지를 알려줍니다; 그러나, 이 bit 는 non-expedited 전송에만 사용됩니다.

expedited transfer 에서, 한편으로, 사용자 데이터는 이러한 최종 4 byte 내에서 직접 전송됩니다. 프로토콜 바이트의 추가 2 bit 는 이러한 byte 들이 실제로 얼마나 할당되는지를 명시합니다 (사용자 데이터의 단 1 byte 전송도 가능합니다). 따라서 사용자 데이터는 반드시 SDO 오브젝트의 데이터 영역에서 왼쪽-정렬에 위치해있어야 합니다.

일반적으로 CANopen 에서 데이터는 **"Little Endian"**규칙, 즉 관련 INTEL 프로세서의 형태에 따라 전송된다는 것에 유의해야 합니다. 이는 낮은 값의 byte 가 먼저 전송된다는 것을 뜻합니다. 이것은 인간이 측정된 SDO 프로토콜 순서를 따라가는 것을 약간 어렵게 만들지만 결국은 습관화의 문제입니다. 이제까지 설명한 프로토콜 바이트의 bit 를 세어보면, 7 bit 들이 있습니다. 8 번째 bit 는 따로 남겨둡니다.

OD entry [1017]에 대한 SDO 다운로드는, heartbeat producer 의 hearbeat 간격으로, 4 초(in ms as an UNSIGNED16 value, i.e. 0x0F A0)로 설정되므로 다음과 같이 나타납니다:

2B	17 10	00	A0 0F 00 00

그런 후 노드 (SDO 서버)는 메시지로 성공적인 완료를 인지합니다

60	17 10	00	00 00 00 00

초기의 **SDO Upload service 로,** CANopen 노드의 객체 사전 엔트리가 읽혀지면서, 데이터 영역의 동일한 division 이 유효하게 되고, 단 이곳에서의 요청과 응답 정보들은 어느 정도 달라집니다(reverse). 여기서 클라이언트 요청의 명령 바이트는 0x40 입니다:

Command byte (0x40)	OD Main Index	OD Sub Index	Empty (4 bytes)

SDO 서버는 다음으로 응답합니다:

Command byte	OD Main Index	OD Sub Index	Data (4 bytes)

SDO 서버는 디바이스 제조사 (vendor ID; found under subindex 1 of the identity object [1018])를 읽기 위한 일반적인 요청에 항상 응답해야 하는데, 그것은 이 OD-entry 가 강제적인 것이기 때문입니다. 여기의 예에서, 이 디바이스는 IXXAT 사 (vendor no.: 00 00 00 04) 것입니다. 따라서 SDO 응답 메시지는 다음과 같습니다:

43	18 10	01	04 00 00 00

"expedited transfer" 가 사용되지 않는다면, Initiate SDO service 의 4 개 data byte 는 전송될 사용자 데이터의 길이 (byte)를 지정하는데 사용될 수 있습니다. 그러면 실제 전송은, Download SDO Segment 또는 Upload SDO Segment service 와 같이 발생합니다. 세그먼트마다 사용자 데이터의 7byte 가 전송될 수 있습니다. 이러한 서비스들의 command byte 는, 최종 세그먼트를 제외하고, service-ID (command specifier)의 3 bit, 한 개의 toggle bit 와 네 개의 unused bit 를 포함합니다. 또한 세그먼트 사이즈 7 의 배수가 아닌 사용자 데이터를 안전하게 전송하기 위해서 unused byte (6 과 0 사이의 값)의 수는 최종 SDO 세그먼트의 3 bit 에 코딩됩니다. 마지막으로 command byte LSB 는 데이터 전송의 끝을 표시합니다. 세그먼트 순서는 SDO 요청과 SDO 응답 메시지가 순환적으로 변환(toggle)되는 toggle bit 에 의해 관찰됩니다. 아래에 non-expedited segmented SDO upload 의 코멘트 순차가 설명되어 있습니다.

```
40 08 10 00 00 00 00 00 // Initiate req: Read Device Name [1008]
41 08 10 00 1A 00 00 00 // Initiate resp: Fine. It's 26 bytes long
60 00 00 00 00 00 00 00 // Upload segment req, Toggle = 0
00 54 69 6E 79 20 4E 6F // Upload segment resp, Toggle = 0
70 00 00 00 00 00 00 00 // Upload segment req, Toggle = 1
10 64 65 20 2D 20 4D 65 // Upload segment resp, Toggle = 1
```

60 00 00 00 00 00 00 00 // Upload segment req, Toggle = 0
00 67 61 20 44 6F 6D 61 // Upload segment resp, Toggle = 0
70 00 00 00 00 00 00 00 // Upload segment req, Toggle = 1
15 69 6E 73 20 21 00 00 // Last segment, 2 bytes free, Toggle = 1

CANopen specification DS-301 의 버전 4 에서는, 새로운, 상당히 더욱 효과적이면서도 한층 정교해진 SDO 모드가 소개되어 있는데, 그것이 바로 **SDO block transfer** 입니다.
위에서 설명한 세그먼트 전송과는 대조적으로, 여기서의 세그먼트들은 더 이상 개별적으로 인지되지 않으며, 함께 블록으로 들어가는데, 이것들은 하나씩 각각 전송됩니다. 그러면 파트너가 그 블록을 인지합니다. 29 byte 의 사용자 데이터 크기에서, 프로토콜 오버헤드 관점에서는 블록 전송이 더 좋습니다. 블록 전송에서, 프로토콜 바이트는 각 블록의 개별 세그먼트의 수를 세는데, 블록당 최대 127 세그먼트가 가능합니다. 전송은 초기(initialization) 단계에서, 양쪽 파트너들의 유틸리티 데이터 크기와 블록 사이즈가 서로 맞게 만들어지고, 종료(termination) 단계에서, 초기화 동안에 통신 파트너들이 이에 동의했다면, 전체 전송에 관한 CRC check sum 이 기록됩니다. 그러나 SDO 블록 전송은 현재 몇몇 디바이스들만 지원하고 있습니다.

객체 사전 엔트리 [1008]에 대한 SDO read 액세스, 디바이스 이름은, 블록 전송이므로 다음과 같습니다:

A4 08 10 00 21 00 00 00 // Initiate req: Read [1008], Blocksize 33, CRC supported
C2 08 10 00 1A 00 00 00 // Initiate resp: It's 26 bytes long, no CRC
A3 00 00 00 00 00 00 00 // Initiate block req: Let's go
01 54 69 6E 79 20 4E 6F // Upload block resp, Segment = 1
02 64 65 20 2D 20 4D 65 // Upload block resp, Segment = 2
03 67 61 20 44 6F 6D 61 // Upload block resp, Segment = 3
84 69 6E 73 20 21 00 00 // Last segment, Segment = 4
A2 04 21 00 00 00 00 00 // Upload block req: 4 segments received, Blocksize 33
C9 00 00 00 00 00 00 00 // End resp: 2 bytes free
A1 00 00 00 00 00 00 00 // End req: Thank you

중요한 SDO 서비스는 "Abort SDO Transfer" (command byte: 0x80) 입니다. 이는 정확히 한 개의 CAN 메시지로 구성되며, 정기적인 SDO 프로토콜 메시지 대신, 두 개의 파트너들 가운데 어느 쪽에 의해서도 아무때나 전송될 수 있으며 SDO 전송의 즉각적인 종료를 가져옵니다. 가장 흔한 상황은 주소가 매겨진 객체 사전 엔트리가 존재하지 않을 때 초기 SDO 요청에 대한 응답으로써의 SDO-Abort 입니다. Abort SDO 메시지의 구조는 다음과 같습니다:

0x80	OD main-index	OD sub-index	Abort code

이 SDO 서비스의 데이터 영역은 dword 로 오류 원인 (abort code)을 포함하고 있습니다. CANopen specification 에는 정의되어 있는 모든 abort code 리스트가 있습니다. 총 30 여개 정도입니다. 독자적인 abort code 또는 비-정의된 코드의 사용은 허용되지 않습니다. 가장 중요한 abort code 는 다음과 같습니다.

0x05030000 Toggle bit not alternated
0x05040001 Invalid SDO Command specifier
0x0601000x Unsupported access to an object
0x06010002 Attempt to write a read only object
0x06020000 Object does not exist in the object dictionary
0x0607001x Data type does not match
0x06090011 Subindex does not exist
0x08000000 General error

모든 디바이스는 적어도 한 개의 서버 SDO 채널을 지원해야 합니다. 추가로 더 많은 SDO 채널들이 객체 사전 엔트리(object dictionary entry)를 통해 설정될 수 있습니다. pre-defined record SDO 파라매터를 이용하여 [1201] 부터 [127F] 까지의 OD 엔트리들이 서버 SDO 채널의 정의를 위해 준비되어 있습니다.